Software Architecture for Planetary & Lunar Robotics

Hans Utz
Research Institute for Advanced Computer Sciences at
Intelligent Robotics Group
NASA Ames Research Center

Terry Fong
Intelligent Robotics Group
NASA Ames Research Center

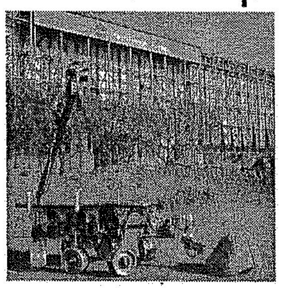
Issa A.D. Nesnas
Jet Propulsion Laboratory
California Institute of Technology

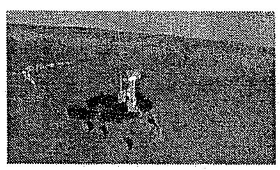
Overview

- The Intelligent Robotics Group
- The Lunar Mission
  - Purpose
  - Time-line
- Lunar Robotics
  - How robots can support the lunar mission
  - What challenges they have to meet
- Software Architecture for Space Robotics
  - CLARAty
  - Meeting the lunar challenges

Intelligent Robotics Group

- Areas of expertise
  - Applied computer vision
  - Human-robot interaction
  - Interactive 3D visualization
  - Robot software architectures
- Science-driven exploration
  - Survey, instrument placement, resource mapping
  - Low speed, deliberative operation
- Fieldwork-driven operations
  - Assembly, inspection, maintenance
  - Pre-cursor missions (site preparation, infrastructure emplacement, etc.)
  - Manned missions (human-paced interaction, peer-to-peer assistance)





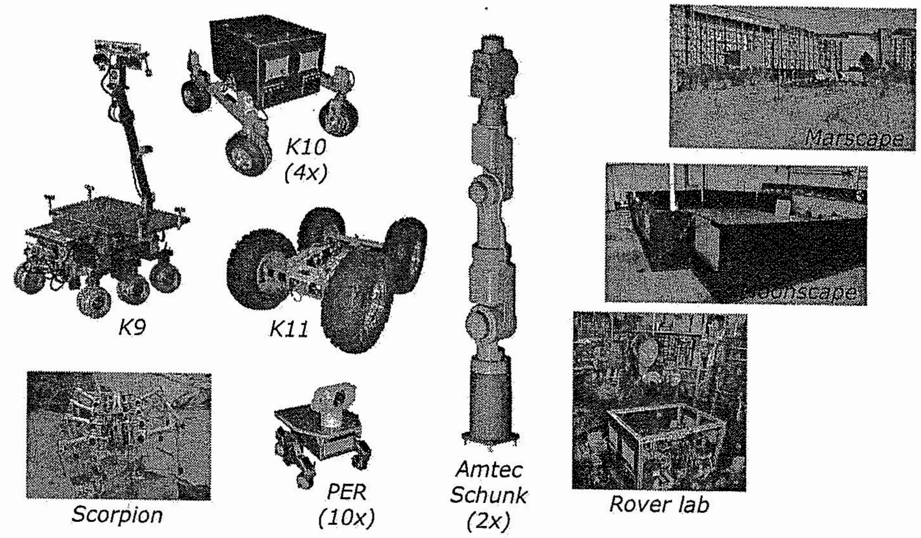
Intelligent Robotics Group

- Group lead: Terry Fong
- Permanent staff: 20 people
- Interns: 5 to 25

Current Activities

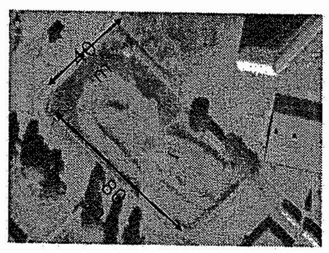
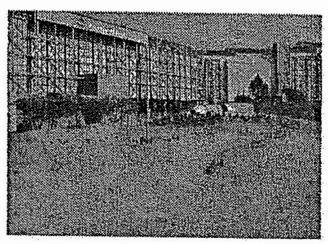
- Missions
  - MER: Viz, MI Toolkit
  - MSL: Viz Explorer (Ensemble)
- Science (SMD)
  - ASTEP: Life in the Atacama, Astrobiology Machine Vision Toolkit
  - Mars Technology Program: K9 rover testbed, SCIP
  - Mars Critical Data Products: MOC image processing (Malin)
- Exploration (ESMD)
  - Human-System Interaction, Surface Handling, and Surface Mobility Systems (ESAS 12BCE)
- Non-NASA
  - Global Connection (CMU - Google - National Geographic)

Robots and Facilities



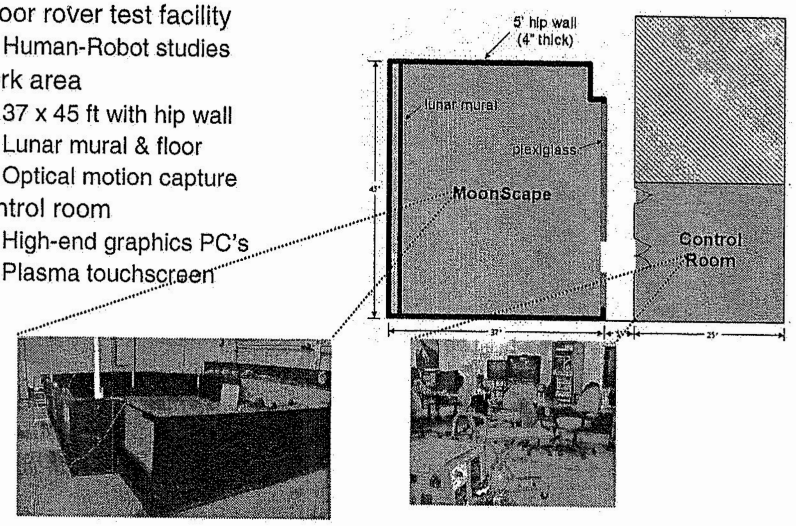
Marscape

- Outdoor rover test facility
  - 3/4 acre, surveyed site
  - Operations trailer
  - dGPS, wireless LAN, power
- Mars analog
  - Design reflects geology of scientific interest
  - Streambed, delta, lakebed, volcano, chaotic terrain, meteorite impact crater, etc.
  - Traversable + non-traversable regions (with occlusions)

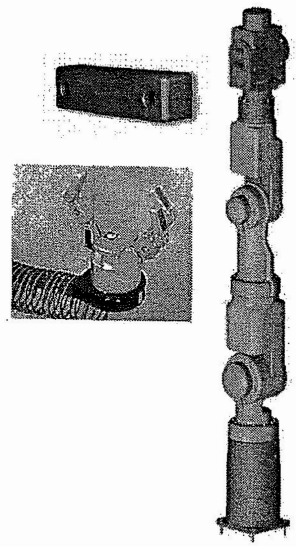



Moonscape

- Indoor rover test facility
  - Human-Robot studies
- Work area
  - 37 x 45 ft with hip wall
  - Lunar mural & floor
  - Optical motion capture
- Control room
  - High-end graphics PC's
  - Plasma touchscreen

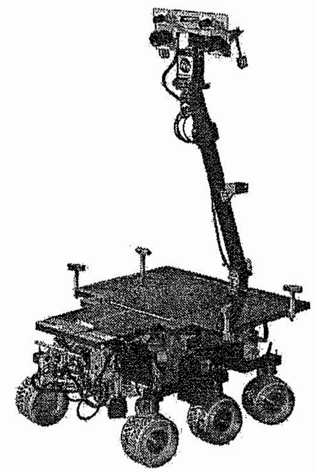


Arm Lab



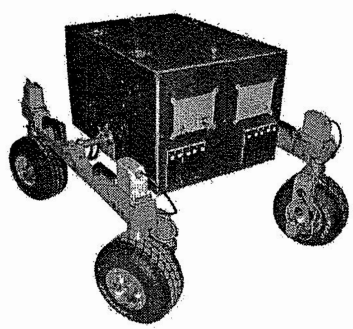
- Dexterous Manipulation
  - Force and vision-guided manipulation
  - Science sampling (rock collection)
  - Surface ops (mobile manipulation)
- Equipment
  - 2x Amtec Schunk Ultra Light Weight arms
    (7-DOF, 1m reach, 1.5kg payload)
  - 2x Barrett 3-finger hands
  - Stereo vision (PointGrey Bumblebee)
  - Force/torque sensors (wrist & fingers)
- Development
  - 2006: testbed completion & initial API
  - 2007: integration on K10 rover

K9 Rover



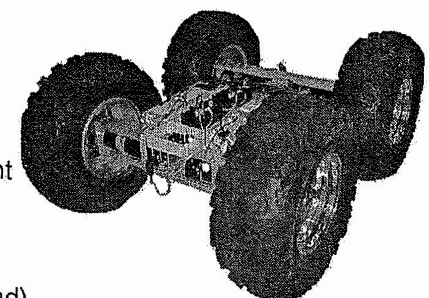
- Planetary science rover
  - Remote autonomous experiments
  - In-situ measuements
- Characteristics
  - FIDO chassis: 6-wheel steer rocker-bogey
  - 5-DOF instrument arm
  - Size: 1.7 x 0.8 x 1m (HxWxL) with mast
  - Speed: 6 cm/s
  - Power: 570 W (Li-Ion batteries)
  - Weight: 70 kg
- Instrumentation
  - CHAMP: Camera, Handlens, and Microscope Probe (Mungus / JPL)
  - 6x Dragonfly cameras (navigation)
  - 2x Basler area scan cameras (science)

K10 Rover



- Field work rover
  - Operational tasks (assembly, inspection, etc.)
  - Human paced operations
  - Same avionics and software as K9
- Characteristics
  - 4-wheel steer rocker chassis
  - Low-cost (COTS parts)
  - Size: 0.6 x 0.7 x 1 m (HxWxL)
  - Speed: 0.8 m/s (10 deg slope)
  - Power: 1900 W (Li-Ion batteries)
  - Weight: 100 kg (30 kg payload)
- Development
  - 2004: initial build (Hsiu / Gogoco LLC)
  - 2005: rev. 2 build
  - 2006: locomotion redesign (Proto Innovations LLC)

K11 Rover



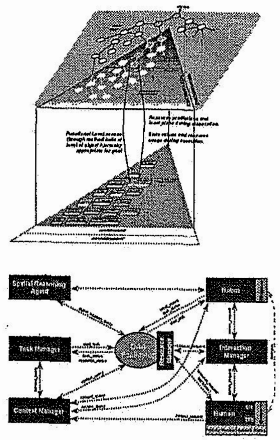
- Extreme environment rover
  - Power-efficient mobility
  - All-terrain capability
  - Antarctic exploration
- Characteristics
  - Skid-steered with body roll joint
  - Size: 1.5 x 1.13 x 0.65 m
  - Speed: 1 m/s
  - Power: ~200 W (solar)
  - Weight: 250 kg (100 kg payload)
- Development
  - Collaboration with EPFL Autonomous Systems Lab and BlueBotics SA
  - 2005: trade studies & component tests
  - 2006: mechanical prototype

Research Areas

Perception

Interaction

Architecture



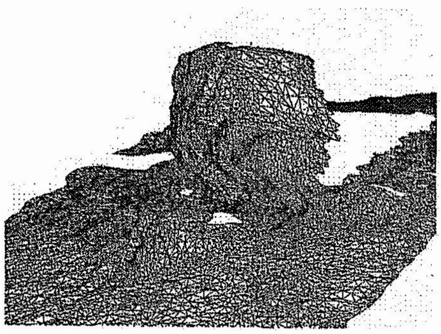
Perception

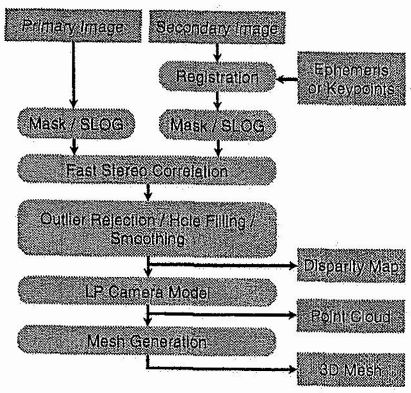
- Science
  - Systematic data collection
  - Surface reconstruction
  - Scientific image processing
- Navigation
  - Local rover navigation (approach and return to target)
  - Single-cycle instrument placement

M. Broxton, M. Deans, S. Desiano, L. Edwards, C. Kunz, L. Pedersen

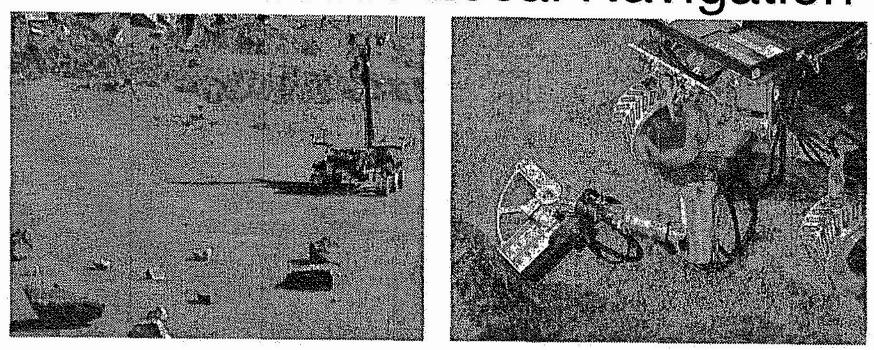
Surface Reconstruction

- Terrain Modeling with VisionWorkbench
  - High-quality meshes / DEM built from orbiter & rover camera data
  - Supports mission planning and science (ground control)
  - Assists rover navigation and collision avoidance (on-board)





Vision-based Local Navigation



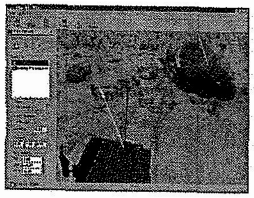
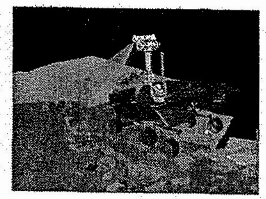
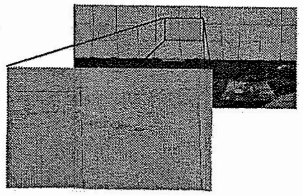
Navigation
- Multiple targets / cameras
- 10m range
- 1cm accuracy

Hand-off
(Multi-camera tracking)

Placement
- Rock segmentation
- Instrument safety
- Arm motion planning

Interaction

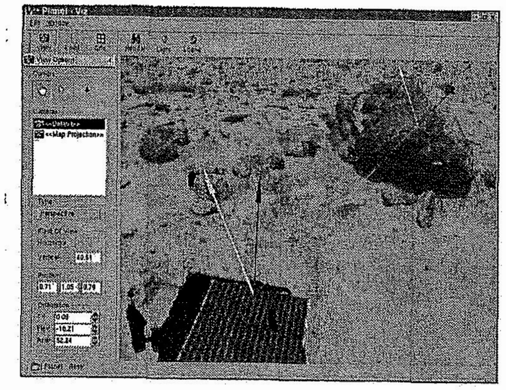
- Peer-to-Peer HRI
  - Human-robot teaming: dialogue and coordination
- Viz
  - Simulation and scientific visualization
  - Mission ops (Ensemble integration)
- Global Connection
  - Giga-pixel panoramas (high dynamic range & time-lapse)
  - Disaster response, education & community building

L. Edwards, T. Fong, D. Lees, L. Keely, C. Kunz, R. Sargent

Viz

- Interactive 3D UI
  - Stereo viewing
  - Background image
- Simulation
  - Time of day lighting
  - Viewpoint + pose
  - Object kinematics
- Site understanding tools
  - Point, distance, azimuth measure
  - Elevation + slope maps
  - Sun + planet vectors
  - Surface area measure
  - Terrain cross-section
  - Markers + ancillary data



Missions: MPL, MER, Phoenix, MSL
Field Tests: K9/Mojave, IS Level 1, CDS

Global Connection

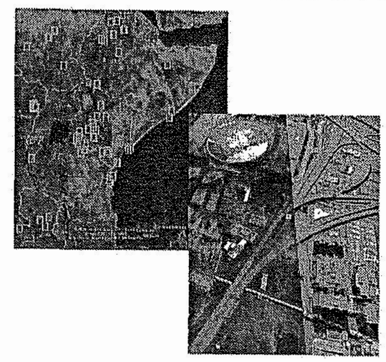
- Connecting through advanced imaging
  - Building local and global communities
  - High performance capture and display
  - High resolution images: spatial – temporal – dynamic range
- Public / private collaboration
  - PI's: Randy Sargent and Illah Nourbakhsh
  - NASA Ames Research Center, CMU Robotics Institute, Google Earth, National Geographic
  - `http://www.cs.cmu.edu/~globalconn`

Global Connection

THE ROBOTICS INSTITUTE Google NATIONAL GEOGRAPHIC

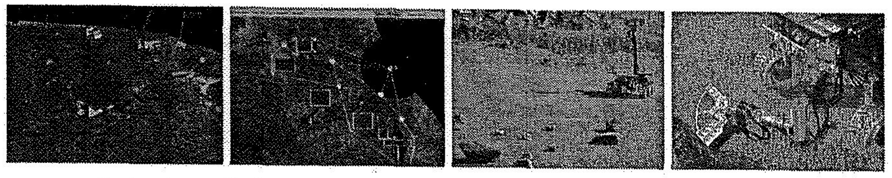
Global Connection

- Spatial Browsing
  - Geo-referenced aerial images & hypermedia
  - For education and entertainment
  - Collaboration with Google & National Geographic
- Disaster Response
  - Rapid geo-referencing of aerial & satellite images
  - For improved planning and logistics
  - Collaboration with Google & NOAA
- Gigapixel Panoramas
  - Ultra high-definition capture and interactive display
  - For education, entertainment, and science




Architecture

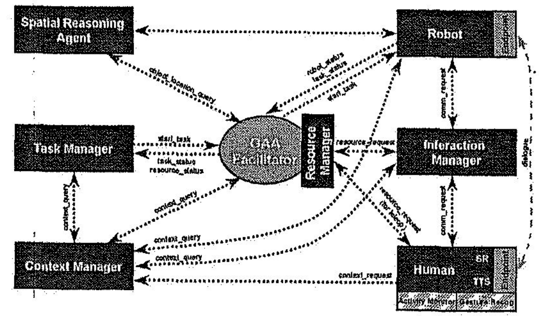
- CLARAty
  - Component-based software engineering (for robustness & reuse)
  - Extensions for dynamic environments (support ESMD activities)
- Multi-SCIP
  - Flexible command cycles (goal-level on human time-scale)
  - Fast science autonomy (10-100x increase in science return)
- HRI/OS
  - Human-robot teaming via peer-to-peer dialogue & cognitive models



S. Desiano, T. Fong, L. Fluckiger, C. Kunz, E. Park, L. Pedersen, V. To

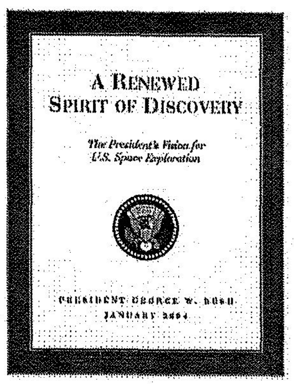
HRI/OS Architecture

- Human-Robot Interaction Operating System
  - Structured interaction framework (not robot controller)
  - Agent-based system design
  - Multiple humans/robots, near/far interaction, adjustable autonomy
- Core services
  - Data distribution
  - Human-robot dialogue
  - Resource management
  - Service delegation
  - Task management
- Extensible API
  - Robots & UI's
  - Java, C++
  - Win2K/XP, Linux



The Vision for U.S. Space Exploration

THE FUNDAMENTAL GOAL OF THIS VISION IS TO ADVANCE U.S. SCIENTIFIC, SECURITY, AND ECONOMIC INTEREST THROUGH A ROBUST SPACE EXPLORATION PROGRAM



Implement a sustained and affordable human and robotic program to explore the solar system and beyond

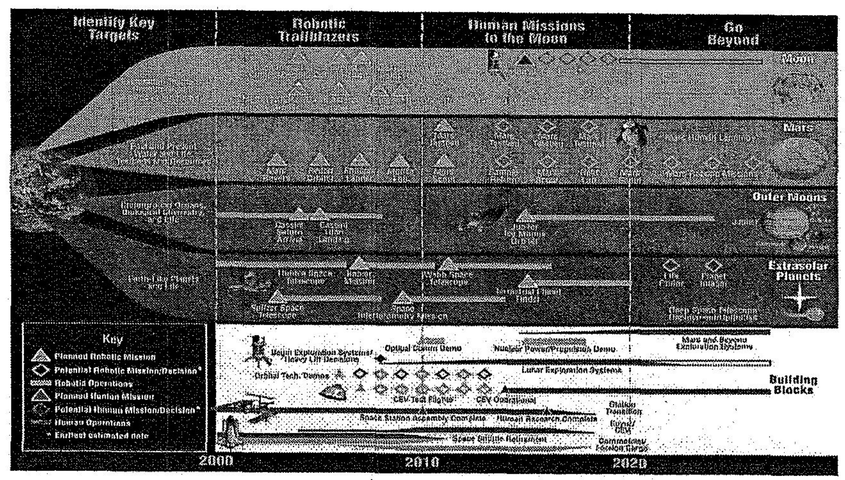
Extend human presence across the solar system, starting with a human return to the Moon by the year 2020, in preparation for human exploration of Mars and other destinations;

Develop the innovative technologies, knowledge, and infrastructures both to explore and to support decisions about the destinations for human exploration; and

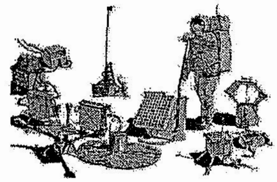
Promote international and commercial participation in exploration to further U.S. scientific, security, and economic interests.

Slides by courtesy of Exploration Systems Mission Directorate

Exploration Roadmap



Exploration Systems Key Objectives & Milestones



- Major Milestones
  - **2008: Initial flight test of CEV**
  - **2008: Launch first lunar robotic orbiter**
  - **2009-2010: Robotic mission to lunar surface**
  - **2011: First CEV flight**
  - **2014: First crewed CEV flight**
  - **2015-2020: First human mission to the Moon**

Space Robotics, Motivation

- Robots are cheap
  - No return ticket
  - No live support system (weight).
- Robots introduce less risk
  - Loss of mission vs. loss of crew
- Robots are more robust
  - Operation in vacuum
  - Radiation
  - Extreme temperatures

- Robots are essential in supporting the human space flight program.

Lunar Robotic Tasks

- Robotic missions can play an important role in supporting all phases of the human space flight program:

1. Landing Site Selection
2. Autonomous Outpost Preparation
3. Human Mission Assistance

Landing Site Selection

- Collecting data to help answering pending questions such as:
  - Water at the Lunar poles?
  - Suitability for Lunar habitat?

Mission Modes:

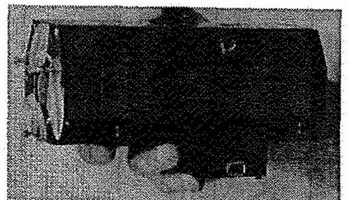
- Geologic site survey
- Landing site inspection

But:

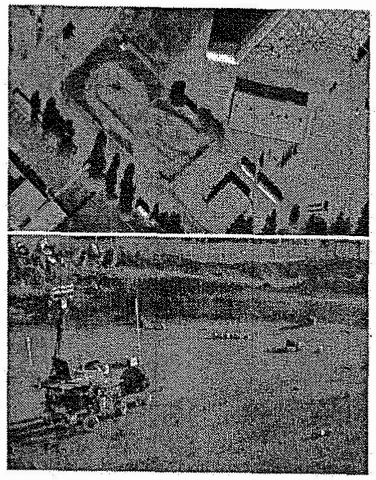
- Orbital imagery covers larger terrain (if sufficient)

Human-Robot Site Survey (2006)

- K9 and K10 in Marscape
  - Comprehensive resource mapping (prospecting)
  - Multiple rovers and HRI
  - DEM from aerial imagery (+ surveyed control points)
  - Transect of traversable zones
  - Surface (CHAMP) and sub-surface sampling (MUM)



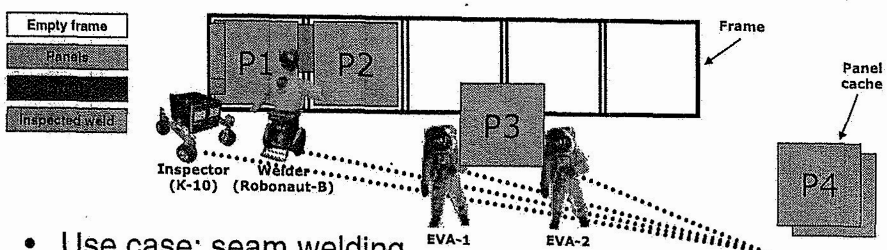
Camera, HAndlens and Microscope Probe (JPL, Mungas)



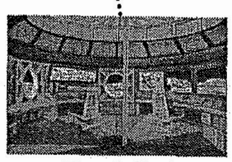
Autonomous Outpost Assembly

- The ESAS study model of outpost assembly requires minimal robotic assistance
  - Wiring the outpost with the power plant
- Assembling the habitat on the Moon can further save weight (launch cost reduction)
- Landing the habitat separate from the astronauts introduces severe additional risk.

P2P-HRI Study (Nov 2005)



- Use case: seam welding
  - Construction of work hangars, shelters, etc.
  - Contingency handling (e.g., repair rupture or break)
- Multi-agent team
  - (1x) IVA astronaut + (2x) EVA astronauts
  - (2x) Robots: "welder" + "inspector"
- Peer-to-peer human-robot interaction
  - Robots ask questions ("Is this weld okay here?")
  - Spatial dialogue ("Shine the light to my left")



IVA-1 (habitat)

Human Mission Assistance

- Extra Vehicular Activity (EVA) is high risk
- Limited EVA time per mission
- Enhancing efficiency of EVA operations has high benefit
- Automating maintenance tasks leaves more EVA time for science

Robot Software Architectures

Almost no SW-reuse in flight software so far

Rising demand for reusable robotics infrastructure:
- The rising complexity of scenarios does not allow reimplementation from scratch
- Complexity of future applications goes beyond the scope of a single research group
- Code reuse enhances software quality
- Code reuse allows focusing on the not yet solved problems

Middleware for Space Robotics?

- Middleware does hardly meet today's requirements of flight software and hardware
  - MHz, MB RAM
  - Software verification: No dynamic memory allocation, no callbacks (virtual methods), no large external libraries
- CLARAty-based demonstrators were flight hardened for use on MER-rovers ?!

CLARAty

Coupled Layer Architecture for Robotic Autonomy

CLARAty is a unified and reusable **robotic software** that provides basic functionality and simplifies the integration of new technologies on various rovers and robotic platforms

- Multi-center project
  JPL, ARC, CMU
- Supports various rover platforms
  Fido, Rocky 7, Rocky 8, K9, K10, (K11)

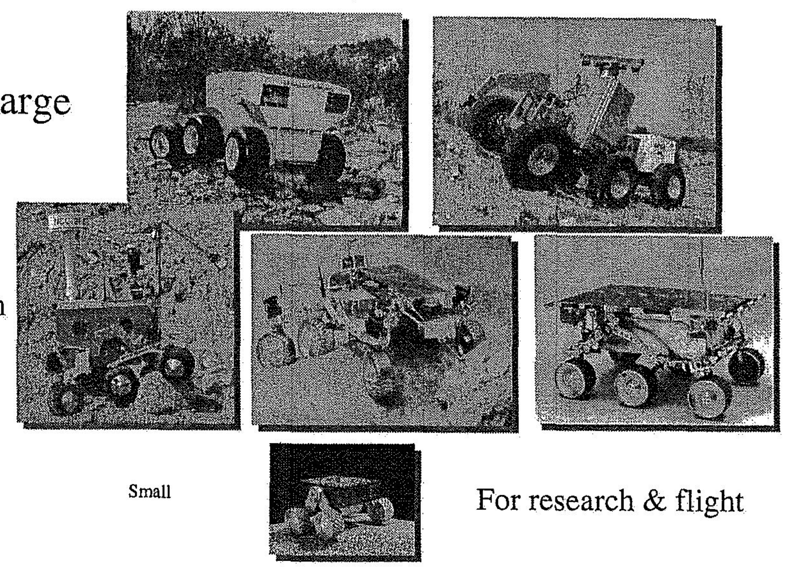
http://claraty.jpl.nasa.gov

Slides in co-operation with Issa Nesnas, JPL

NASA Develops Various Rovers

Large

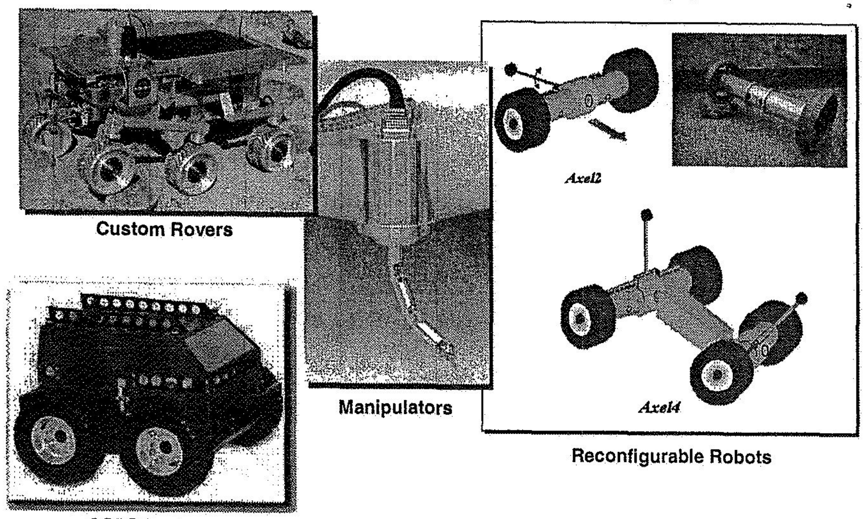
Medium



Small

For research & flight

Would like to support ...



Custom Rovers

Manipulators

Reconfigurable Robots

COTS Systems

Problem and Approach

- Problem:
  - Difficult to share software/algorithms across systems
  - Different hardware/software infrastructure
  - No standard protocols and APIs
  - No flexible code base of robotic capabilities
- Objectives
  - Unify robotic infrastructure and framework
  - Capture and integrate legacy algorithms
  - Simplify integration of new technology
  - Operate heterogeneous robots
  - Mediate between research and flight requirements

Why is robotic software "hard"?

- Software:
  - Software is large and complex
  - Has lots of diverse functionality
  - Integrates many disciplines
  - Requires real-time runtime models
  - Has limited hardware resources - efficiency
  - Talks to hardware
- Hardware:
  - Physical and mechanics vary
  - Electrical hardware architecture changes
  - Hardware component capabilities vary

Why develop reusable software?

- To capture robotic domain knowledge
- To support development of generic algorithms
- To reduce the need for resolving recurring problems for every system
- To simplify integration of new technologies
- To use same framework for various robots
- Increase functionality by leveraging a more mature base

How?

- Study several robotic system implementations
- Study interactions of elements in various systems
- Identify reusable elements in robotic systems
- Identify implicit assumptions made
- Project potential advances to these elements
- Design a generic/flexible implementation of these elements
- Adapt to a number of robotic systems
- Test and study the limitations of the design
- Go back to design and iterate
- Modify/extend/redesign to address limitations and variability across systems

Your generic base is reusable

Approach

- Domain knowledge guides design
- Layers of abstraction help master complexity
- Abstractions also provide a classification of various technology elements
- Information hiding protects implementation variability
- Small modular components are more reusable than monolithic blocks
- Interfaces define behavior of various elements

Things to be aware of

- Over-generalizing leads to ineffectiveness
  - More general -> less functionality -> more work for results
  - Number of abstractions vs. complex hierarchies
  - Modular elements with strongly typed interfaces
  - Algorithm generality influences abstraction design
- Runtime models vary across systems
  - Challenges in combining hardware/firmware/software architectures in most effective manner
  - Need for both cooperative and pre-emptive scheduling

Goals

- Capture and integrate a wide range of technologies
- Leverage existing tools
- Leverage experience and tools of the larger software development community
- Apply appropriate design patterns to the domain
- Provide an infrastructure that enables rapid robotic development
- Capture experience of technologists implementations

Challenges in Interoperability

- Mechanisms and Sensors
- Hardware Architecture

Different Mobility Mechanisms

with different sensors

From wheeled Rocker-bogies with different steering

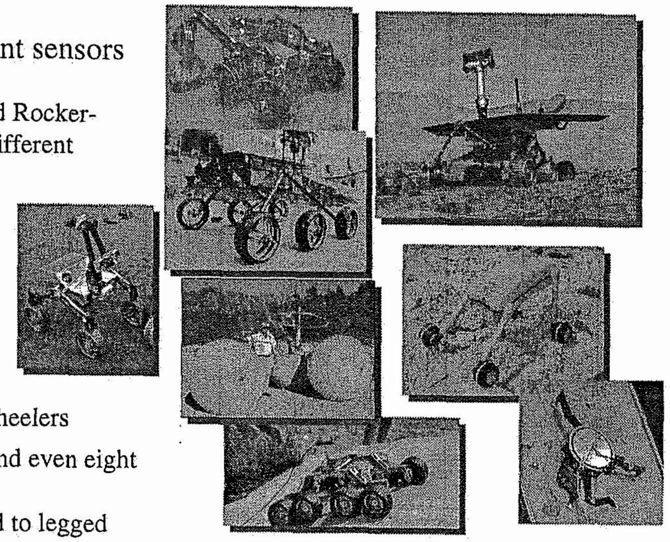
To wheels on articulated links

To inflatable wheels

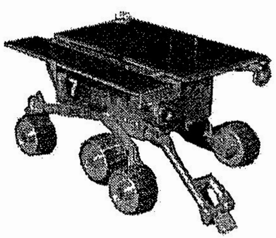
From three wheelers

To four, six and even eight

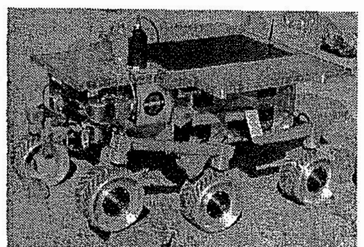
From wheeled to legged



For Example: Wheeled Locomotion



Rocky 7



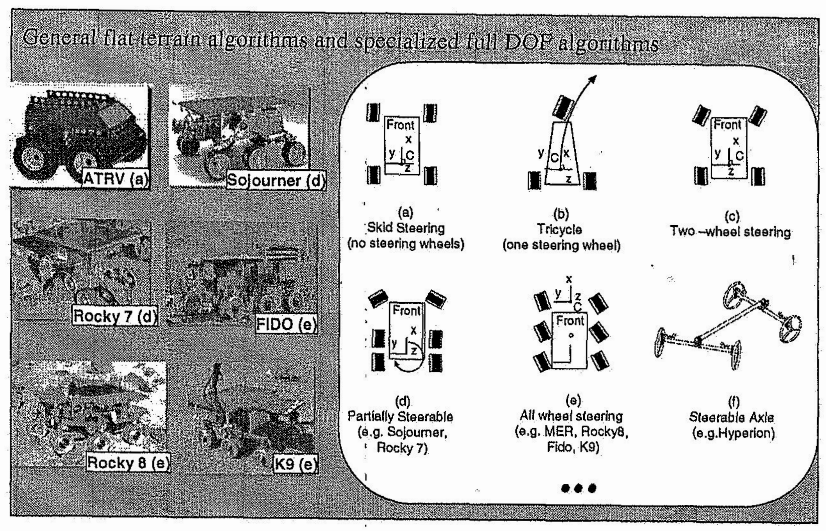
Rocky 8

QuickTime™ and a None decompressor are needed to see this picture.

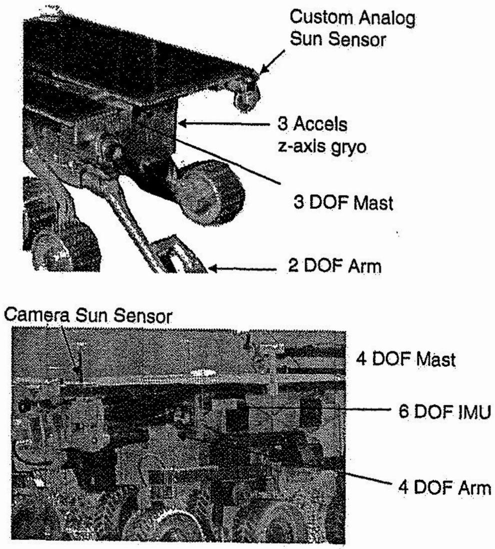
QuickTime™ and a Video decompressor are needed to see this picture.

QuickTime™ and a None decompressor are needed to see this picture.

Reusable Wheeled Locomotion Algorithms



Manipulators and Sensor Suites



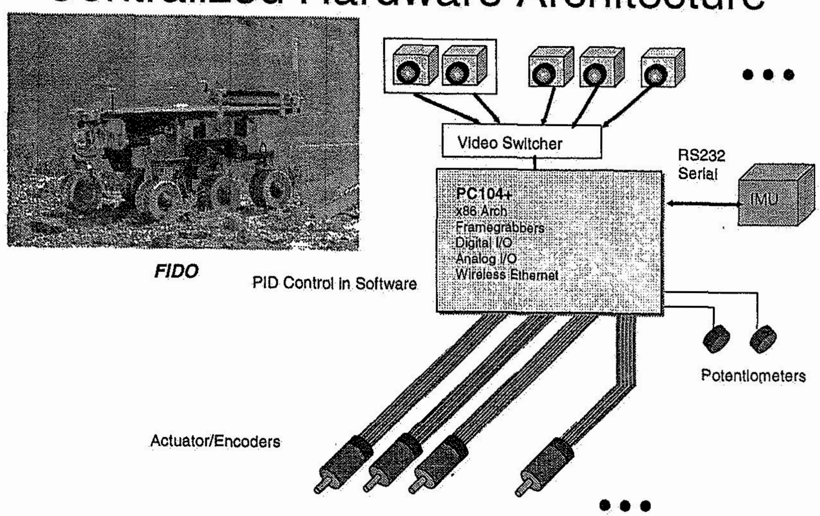
QuickTime™ and a None decompressor are needed to see this picture.

- Given different capabilities, how much reuse can be achieved?

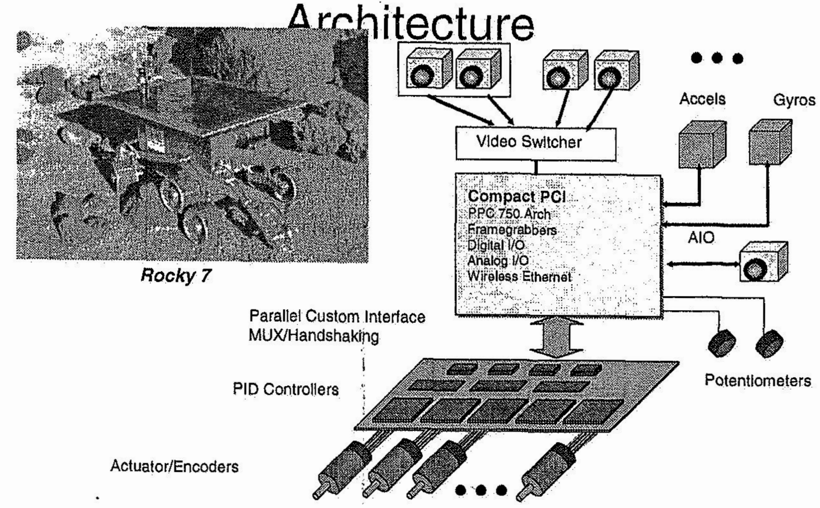
Challenges in Interoperability

- Mechanisms and Sensors
- Hardware Architecture

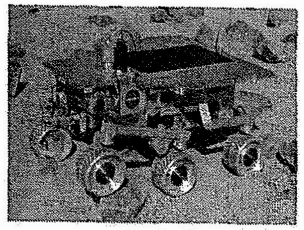
Centralized Hardware Architecture



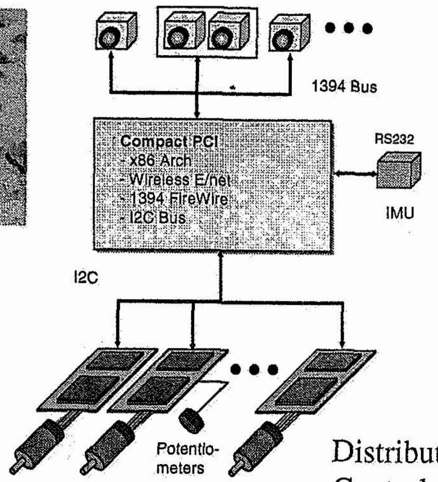
Semi-centralized Hardware Architecture



Semi-distributed Hardware Architecture



Rocky 8



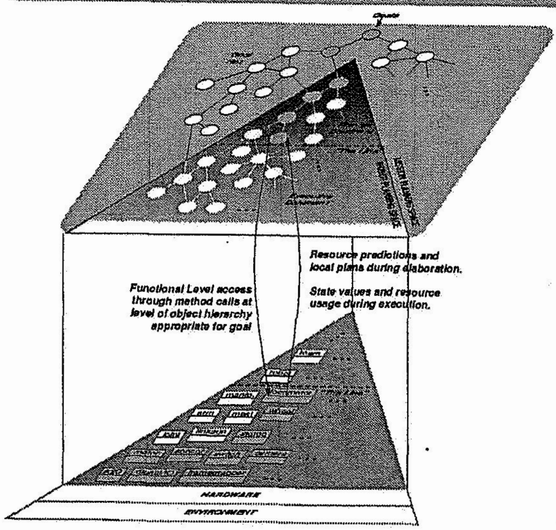
Rocky Widgets
Single-axis controllers
Current sensing
Digital I/O
Analog I/O

Distributed Motion
Control and Vision

CLARAty Architecture

A Two-Layered Architecture

CLARAty = Coupled Layer Architecture for Robotic Autonomy



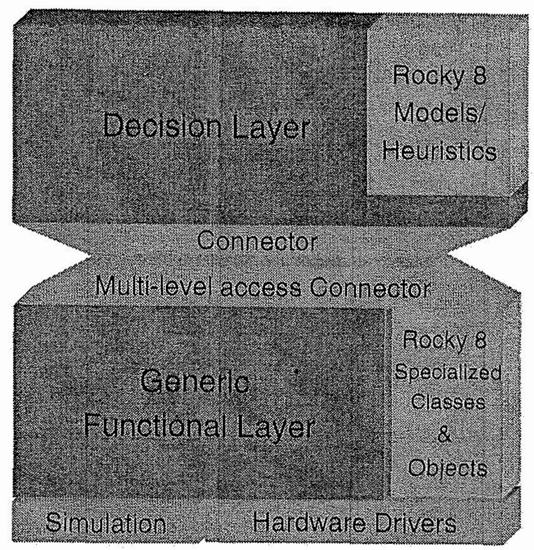
THE DECISION LAYER:
Declarative model-based
Mission and system constraints
Global planning

INTERFACE:
Access to various levels
Commanding and updates

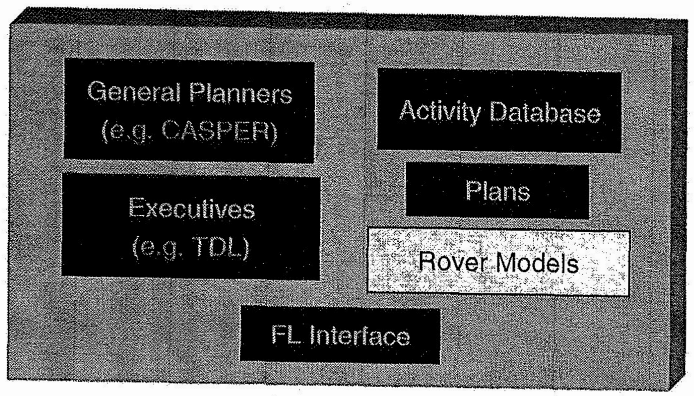
THE FUNCTIONAL LAYER:
Object-oriented abstractions
Autonomous behavior
Basic system functionality

Adaptation to a system

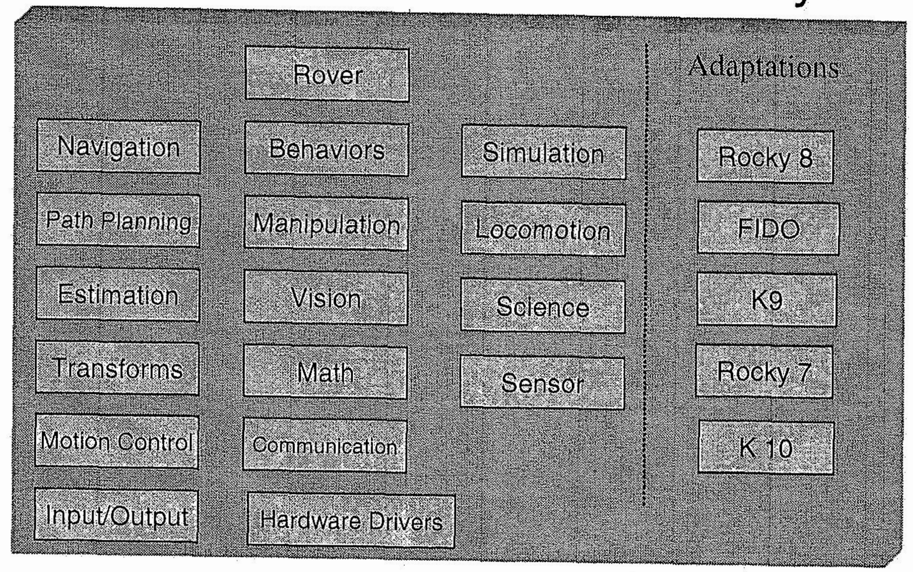
Adapting to a Rover



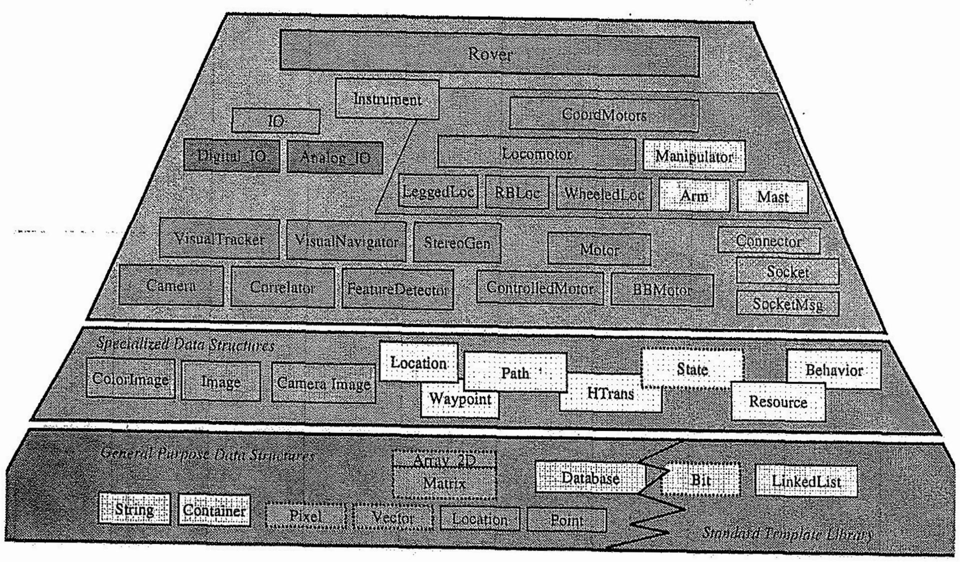
The Decision Layer



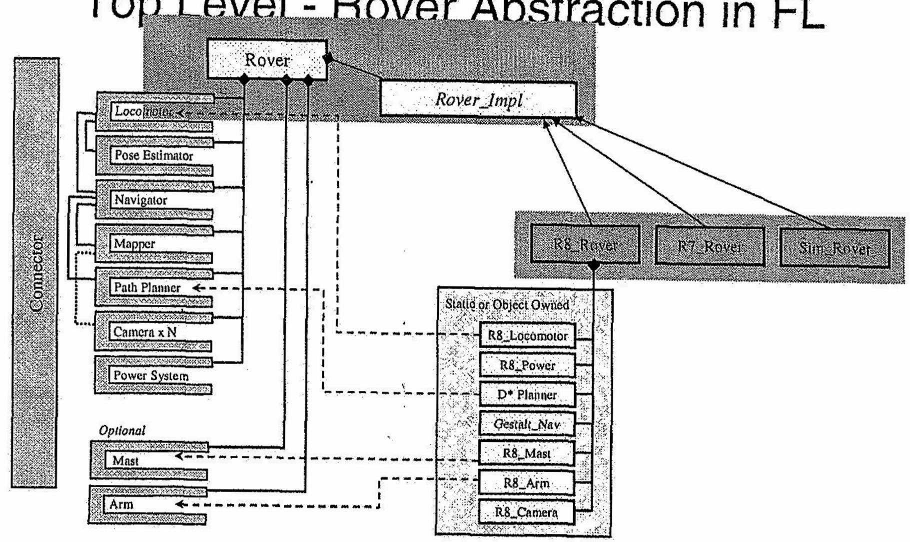
The Functional Layer



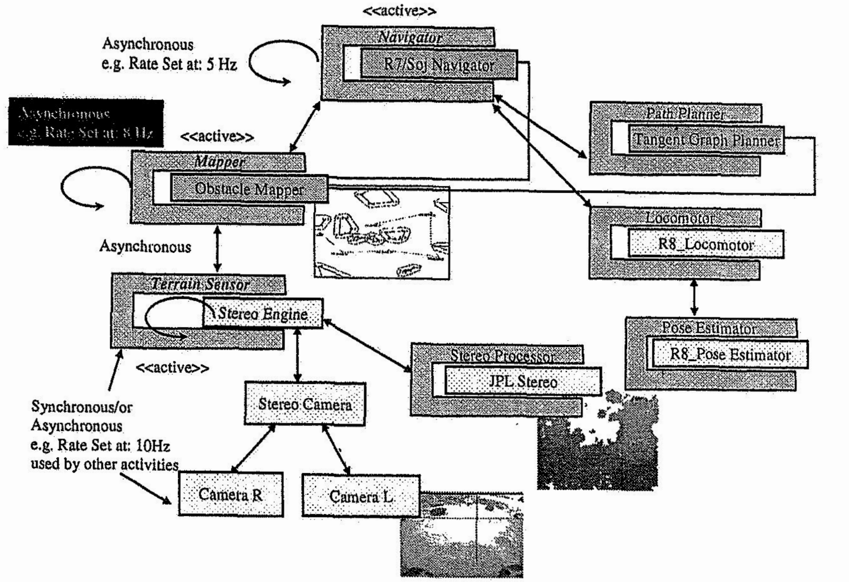
Functional Layer Components



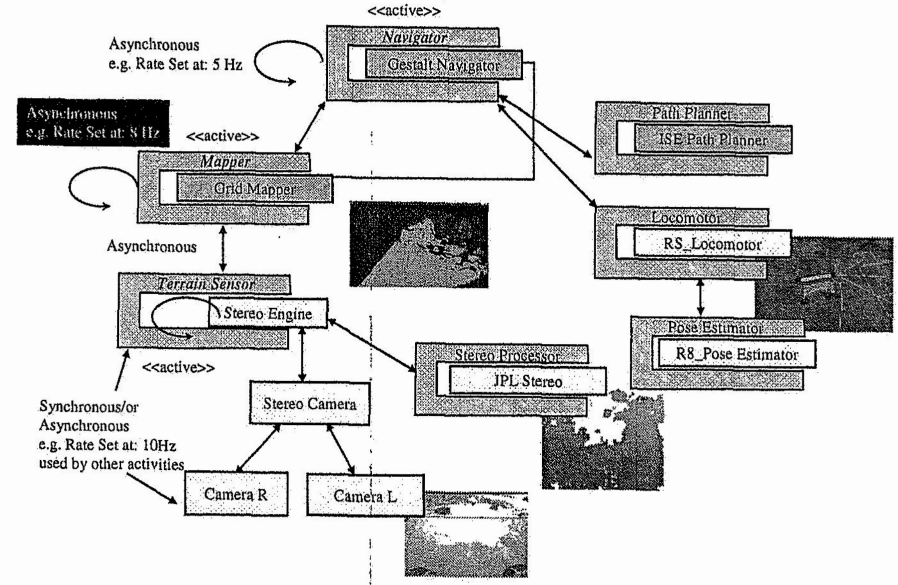
Top Level - Rover Abstraction in FL



Example of Navigation Architecture



Example of Navigation Architecture

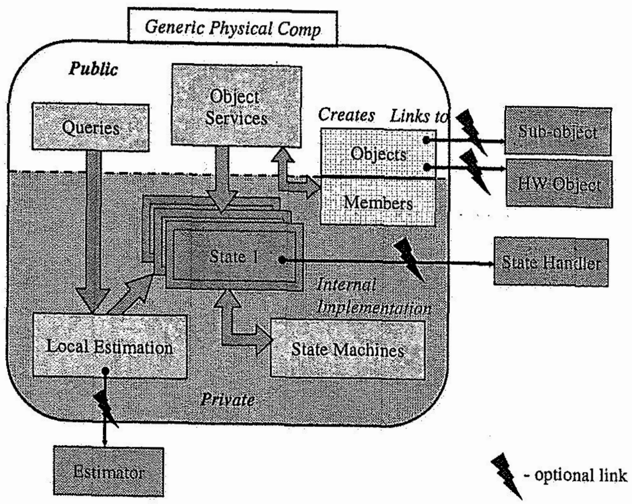


Abstraction Models

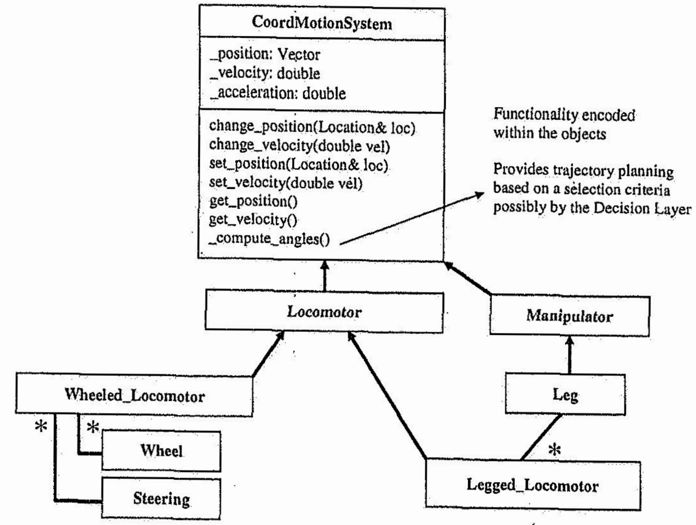
CLARAty Abstractions

- Data Structure Components
  - Array, Vector, Matrix, Map, Container, LinkedList, Bit
  - Image, Message, Resource
- Generic Physical Components (GPC)
  - Locomotor, Arm, Mast,
- Specialized Physical Components (SPC)
  - K9_Locomotor, K9_Arm, R8_Mast
- Generic Functional Components (GFC)
  - ObjectFinder, VisualNavigator, Stereovision, Localizer
- Specialized Functional Components (SFC)

Component Analysis



Relationships with Other Components



Generic Technologies & Algorithms

- Technologies that are generic by design should not be constrained by the software architecture & implementation
- Non-generic technologies should be accommodated on the appropriate platforms
  - Example (Generic): if you are working in navigation, you would not care about H/W architecture difference among different rovers
  - Example (Specific): if you are doing wheel/terrain interaction research, you might require specific hardware which one of the vehicles would support
- Assumptions are made explicit

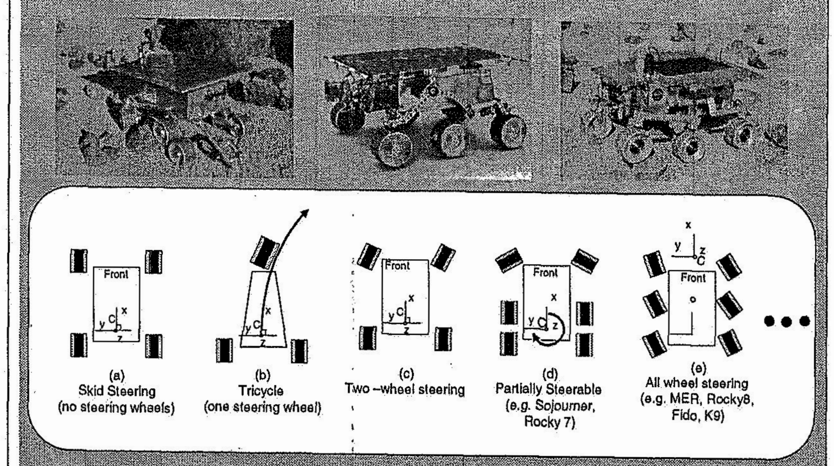
Wheel Locomotor Example

Capabilities of Wheel Locomotor

- Type of maneuvers:
  - Straight line motions (fwd / bkwd)
  - Crab maneuvers
  - Arc maneuvers
  - Arc crab maneuvers
  - Rotate-in-place maneuvers (arc turn r=0)
- Driving Operation
  - Non-blocking drive commands
  - Multi-threaded access to the Wheel_Locomotor class – e.g. one task can use Wheel_Locomotor for driving while the other for position queries
  - Querying capabilities during all modes of operation. Examples include position updates and state queries
  - Built-in rudimentary pose estimation that assumes vehicle follows commanded motion

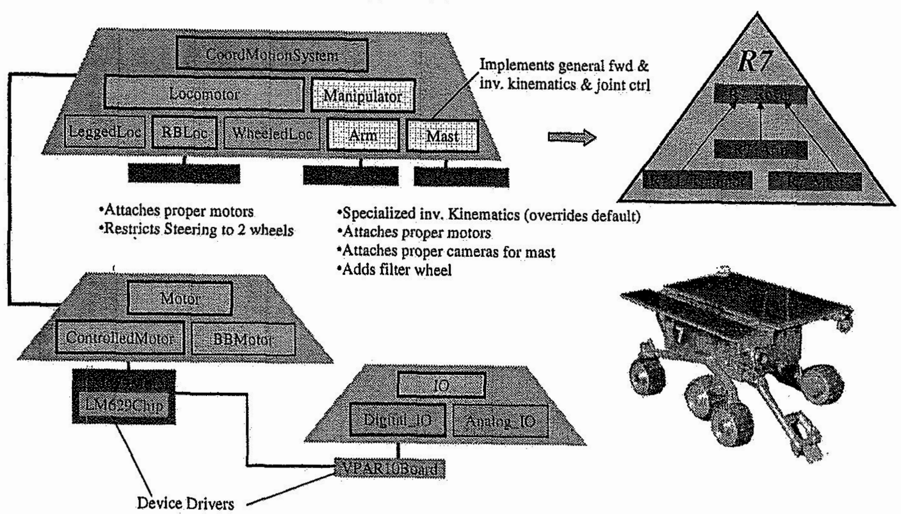
Generic Reusable Algorithms

- Wheeled Locomotion – works for Rocky 8, Rocky 7, Fido, K9, ...



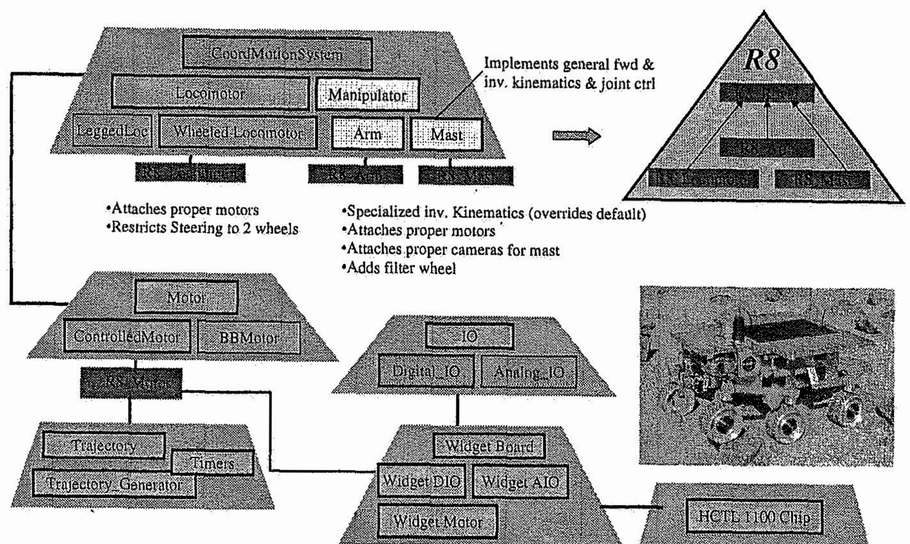
R7 Specific Rover Implementation

Non reusable Code
Reusable Code



R8 Specific Rover Implementation

Non reusable Code
Reusable Code



Code Resuability Results

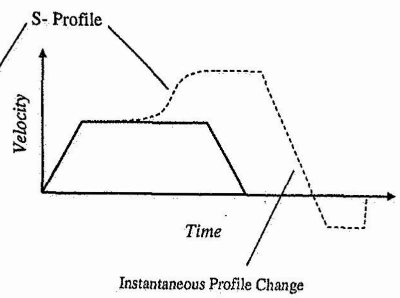
Analysis of amount of resuable code across implementations:

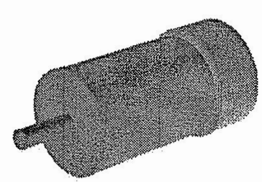
Module	Lines of Code	Status	Depends On
Wheel Locomotor	1445	Reusable	Motion Sequence, 1D Solver, Homogeneous Transforms
Motion Sequence	540	Reusable	Vector
Matrix, Vector, Array	1083	Reusable	-
1D Solver	356	Reusable	-
Location, Homogeneous Transforms	341	Reusable	Rotation Matrix, Point 2D
Rotation Matrices	435	Reusable	-
Point 2D	131	Reusable	-
Controlled Motor	2080	Reusable	
Rocky 8 Locomotor	250	Non-reusable	Rocky 8 Motor
Rocky 8 Motor	334	Non-reusable	Widget Motor, etc...
Total	6995	584 (non-reusable)	
Total Reusable	~92%		

Motor Example

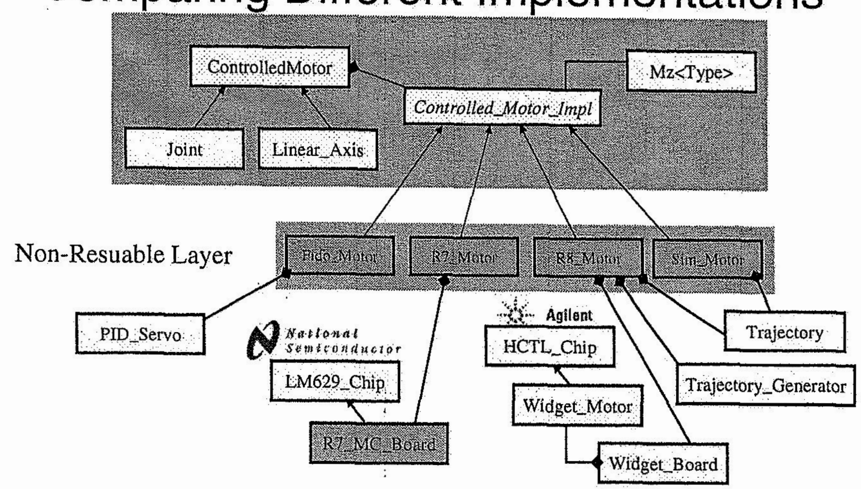
Example: Generic Controlled Motor

- Define generic capabilities independent of hardware
- Provide implementation for generic interfaces to the best capabilities of hardware
- Provide software simulation where hardware support is lacking
- Adapt functionality and interface to particular hardware by specialization inheritance
- Motor Example: public interface command groups:
  - Initialization and Setup
  - Motion and Trajectory
  - Queries
  - Monitors & Diagnostics





Comparing Different Implementations



Modules

- Actual Examples of Code Reusability for Hardware modules:
  - Controlled Motor Hierarchies for Rocky 8 and Rocky7

Rocky 8

Module	Lines of Code	Status
Controlled Motor	2080	Reusable
Trajectory Generator	338	Reusable
Resources (Timers, etc)	143	Reusable
Bits	756	Reusable
Rocky 8 Motor	334	Non-reusable
Widget Motor	383	Reusable –
Motor Controller HCTL	900	Reusable – HCTL
I2C Master	1446	Reusable – I2C
Total	6380	334 (non-reusable)
Total Reusable	~95%	
Total Reusable – Strict	~52%	

Rocky 7

Module	Lines of Code	Status
Controlled Motor	2080	Reusable
Bits	756	Reusable
Input Output	706	Reusable
Rocky 7 Motor	415	Non-reusable
Rocky 7 I/O Maps	131	Non-reusable
Motor Controller LM629	1014	Reusable – LM629
VPAR10 Parallel I/O	534	Reusable – VPAR10
Total	5636	546 (non-reusable)
Total Reusable	~90%	
Total Reusable – Strict	~63%	

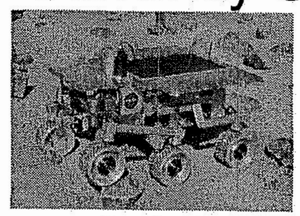
Summary

- CLARAty provides a repository of reusable software components at various abstraction levels
- It attempts at capturing well-known robot technologies in a basic framework for researchers
- It publishes the behavior and interfaces of its components
- It allows researchers to integrate novel technologies at different levels of the architecture
- It is a collaborative effort within the robotics community
- It runs on multiple heterogeneous robots

Future Plans

- Improve current software infra-structure
- Complete development of several packages
- Fully document all packages and technologies
- Provide navigation infra-structure to support and compare different navigation technologies
- Extend packages to fully support Rocky 8, K9, Fido and Rocky 7 rovers
- Provide simulation components at various levels of granularity
- Provide automated means for accessing Functional Layer from Decision Layer.
- Provide richer functionality to the Decision Layer
- Provide resource queries at various levels

Currently Supported Platforms

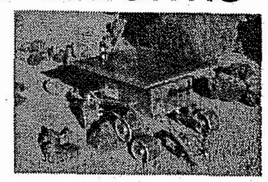


Rocky 8

VxWorks
Intel x86
JPL

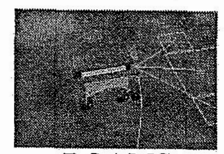
K9

Linux
Intel x86
Ames



Rocky 7

VxWorks
Motorola 68K
JPL



ROAMS

Solaris
JPL

Intel x86
Linux
CMU

Solaris CC
JPL AI

Acknowledgements

CLARAty Team (multi-center)

Jet Propulsion Laboratory
- ROAMS/Darts Team
- CLEaR Team
- Instrument Simulation Team
- Machine Vision Team
- FIDO Team

Ames Research Center
- K9 Team

Carnegie Mellon University

Mars Rovers

CLARAty was developed primarily on Marsian scenarios:
- Communication delay requires high amount of autonomy
- Very limited system resources
- No interaction possible

Scenario limitations:
- Single rover in an otherwise static world
- Limited communication (up/down-link based)
- Low speed requirements
- Limited reactivity

Lunar Challenges

- High Bandwidth, low latency links
  - Extensive communication
  - Limited teleoperation
  - Supervisor feedback instead of autonomy
- Multiple robots
  - Multi-robot cooperation
- Human robot interaction
  - Speed requirements
  - Reactivity requirements
  - Single actor assumption breaks
  - Flexible task allocation

Miro Middleware

- Originated in indoor slam scenario
- Extensively used in RoboCup
- Focus on robot team operations in highly dynamic environments

But:

- Does not care about flight software/hardware limitations

Miro Concepts for Lunar Robotics?

- Open standard based communication infrastructure
  - Type safe, network transparent and programming language independent interfaces
  - Publisher-subscriber architecture for data distribution customized for unreliable wireless networks
- Diverse set of high-level interfaces to robotic services
- Tool supported development model for agile robots in dynamic environments
  - Remote online system inspection
  - High performance logging for data acquisition
  - Online parameterization and re-configuration
- Frameworks for common robotics tasks
  - On board video image processing
  - Reactive behavior based control

Summary

- Space is a challenging application area for mobile robots
- The lunar mission provides new challenges for space robotics
- Software architecture plays an important role in meeting the requirements of future NASA missions.